SIGNAL-01

Avoid using signals or at least keep them short to avoid preemptive interference and reentrancy issues.

Sean Barnum, Cigital, Inc. [vita[1]]

2007-04-16

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 6087 bytes

Attack Category	• Denial of Service • Privilege Exploitation
Vulnerability Category	• Process management • Threading and synchronization problem
Software Context	• Process Management • Debug API
Location	• signal.h
Description	The signal() system call installs a new signal handler for the signal with number signum. The signal handler is set to sighandler, which may be a user-specified function or either SIG_IGN or SIG_DFL. Upon arrival of a signal with number signum the following happens. If the corresponding handler is set to SIG_IGN, then the signal is ignored. If the handler is set to SIG_DFL, then the default action associated with the signal (see signal(7)) occurs. Finally, if the handler is set to a function sighandler, then (1) either the handler is reset to SIG_DFL or an implementation-dependent blocking of the signal is performed and (2) sighandler is called with argument signum. Using a signal handler function for a signal is called "catching the signal." The signals SIGKILL and SIGSTOP cannot be caught or ignored. Because signal calls are preemptive and not necessarily reentrant, there exists the possibility of one signal being called, its handler beginning to execute and then during execution, a second signal being called. In this case, the first signal handler's execution is stopped and the second signal handler executes but when it finishes, It does not necessarily reenter the first signal handler to finish its execution. This leaves the system in an unknown state.
APIs	**Function Name** / **Comments** signal

1. http://buildsecurityin.us-cert.gov/bsi-rules/35-BSI.html (Barnum, Sean)

Method of Attack	An attacker could rapidly call various signals over and over causing preemptive and reentrant conflicts between signal handlers leaving the system in a potentially unstable state.
Exception Criteria	
Solutions	

Solution Applicability	Solution Description	Solution Efficacy
Generally applicable.	Minimize use of signals. They are nonportable and their behavior varies greatly from system to system.	Effective
Generally applicable.	Minimize signal handler functionality to reduce the potential window for overlap between signal calls. Keep signal handlers small and straightforward. A good strategy is to use the signal handler to set a flag of some sort that another part of the program will look for and act on. Do not allow signals to run at privileged level.	Makes exploitation more difficult but does not resolve the vulnerability completely.
Generally applicable.	Encase signal handlers in a self-checking wrapper to ensure that a given handler is nonselfreentrant (i.e., that calling signal1 while they signal1handler is executing	Effective at eliminating risk of selfreentrant signal conflicts but does not effect potential intersignal conflicts.

		does not restart signal1handler but ignores any signal1 calls until the signal1handler finishes execution).	
	Generally applicable.	Warn on instances of signal. If possible, trace which function is being registered as the signal handler. If the function is overly large or complex, increase the level of severity of the warning.	Effective at making developer aware of potential issues.

Signature Details	typedef void (*sighandler_t)(int); sighandler_t signal(int signum, sighandler_t handler);
Examples of Incorrect Code	
Examples of Corrected Code	
Source References	• Rough Auditing Tool for Security (RATS)[2] • signal(2) - Linux man page[3]. • Schilling, Jonathan. C++ and Signal Handling[4].
Recommended Resource	
Discriminant Set	**Operating System**: • UNIX (All) **Languages**: • C • C++

Cigital, Inc. Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. mailto:copyright@cigital.com